COLLEGE MANAGEMENT SYSTEM

A PROJECT REPORT

Submitted by

HARNOOR CHAUHAN [Reg No: RA2211030010069]

DIYA KALRA [Reg No: RA22110300100130]

SHREYAN MUKHERJEE[RA2211030010083]

Under the Guidance of

DR. K A Y A L V I Z H I R

Assistant Professor, Department of Networking and communications

In partial fulfilment of the requirements for the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

with a specialization in CYBER SECURITY

DEPARTMENT OF NETWORKING AND COMMUNICATIONS

COLLEGE OF ENGINEERING AND TECHNOLOGY SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR – 603 203

MARCH 2023

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR – 603 203

BONAFIDE CERTIFICATE

Certified that this B. Tech project report titled **"\COLLEGE MANAGEMENT SYSTEM"** is the bonafide work of Ms. Harnoor Chauhan [Reg. No.: RA221030010069], Ms. Diya Kalra [Reg. No.RA2211030010130] and Mr. Shreyan Mukherjee [Reg No.: RA221030010083] who carried out the project work under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion for this or any other candidate.

DR. KAYALVIZHI R
SUPERVISOR
Assistant Professor
Department of Networking and Communications

DR. ANNAPURANI PANAIYAPAN K
HEAD OF DEPARTMENT
Department of Networking and Communications

SIGNATURE OF INTERNAL EXAMINER

SIGNATURE OF EXTERNAL EXAMINER

S.NO.	*TITLE*	*PAGE NO.*
1.	ABSTRACT	4
2.	INTRODUCTION	5
3.	LITERATURE SURVEY	6
4.	NEED FOR PROJECT	8
5.	FEATURES	10
6.	ER DIAGRAM AND ITS EXPLANATION	12
7.	RELATIONAL TABLE AND EXPLANATION	15
8.	ARCHITECTURE DESIGN	25
9.	WEBPAGE	26

ABSTRACT

Efficiently managing tasks, monitoring workflow among multiple people, and keeping track of the same is a crucial aspect of corporate management work. This project aims to deliver a unified and cost-effective solution that provides real-time task assignment on a web-based software. Our task management system offers users an intuitive interface to navigate, create, edit, delete, and organize their objectives. Simultaneously, these tasks are updated for all other users on the network, empowering them with a comprehensive view of their organization’s workflow. The project's standout feature is the web-based application, allowing secure remote server administration and management. Built with security in mind, the application safeguards against common web vulnerabilities while providing a powerful tool for task analysis and management. The project emphasizes robust design and user-friendly interfaces for a seamless user experience. Users can perform various operations with confidence, including task creation, deletion and editing. The project's implementation adheres to best practices, featuring robust testing, comprehensive code documentation, version control, and alignment with coding standards. It serves as a practical and educational example of creating a web-based interface that enhances productivity and task management while remaining cost-effective

INTRODUCTION

General

In today's dynamic educational landscape, characterized by rapid technological advancements and evolving pedagogical approaches, the effective management of academic institutions has emerged as a critical imperative. At the forefront of this endeavor stands the College Management System (CMS), an indispensable digital infrastructure that not only serves as the backbone of educational institutions but also drives their growth and adaptability in an ever-changing environment. Through its seamless integration of technology into administrative processes, enhancement of communication channels, and provision of a comprehensive suite of tools for faculty and students alike, the CMS assumes a pivotal role in reshaping the operational paradigms of modern colleges and universities.

Our College Management System project is poised to redefine the operational framework of educational institutions by introducing a robust, scalable, and user-friendly platform meticulously tailored to address the unique challenges and requirements of contemporary academia. By leveraging state-of-the-art technology and innovative design principles, our CMS endeavors to transcend traditional boundaries and usher in a new era of efficiency, collaboration, and excellence in education.

Our CMS prioritizes the individualized needs of students by offering a personalized learning experience. Through adaptive learning algorithms and customizable course materials, students can engage with content at their own pace and proficiency level. Additionally, the system tracks student progress and provides personalized recommendations for academic support, fostering a nurturing environment conducive to student success.

Recognizing the diverse needs of the educational community, our CMS is designed with accessibility and inclusivity in mind. From providing multi-modal content formats to integrating assistive technologies, such as screen readers and captioning services, the platform ensures that all students, regardless of their abilities or backgrounds, have equal access to educational resources and opportunities.

Leveraging state-of-the-art technology and innovative design principles, our CMS transcends traditional boundaries to usher in a new era of efficiency, collaboration, and excellence in education. With its robust architecture and scalable infrastructure, the platform is equipped to accommodate the evolving needs and aspirations of educational institutions of all sizes and scopes. Whether it's facilitating online learning initiatives, optimizing resource allocation, or providing actionable insights through advanced analytics, our CMS harnesses the power of technology to drive transformative impact and propel educational institutions towards unprecedented levels of success.

LITERATUTE SURVEY

1. R. C. Pushpaleela, S. Sankar, K. Viswanathan and S. A. Kumar, "Application Modernization Strategies for AWS Cloud," 2022 1st International Conference on Computational Science and Technology (ICCST), CHENNAI, India, 2022, pp. 108-110, doi: 10.1109/ICCST55948.2022.10040356. keywords: {Cloud computing;Automation;Web services;Scientific computing;Digital transformation;Transforms;Containers;AWS(Amazon Web Service) Cloud;AWS Farget;Server less;CICD;Jenkins;Cloud Computing;Cloud Migration;Cloud Hosting;IaaS & PaaS;IaC;terraform;ECR (Elastic Container Service)},

In the IT business, cloud computing has recently gained a lot of attention. II businesses are considering embracing the cloud since it offers a simple, affordable method of hosting apps and dynamically scaling them. The purpose of this research paper is to study and discuss about Modernization strategies for the digital transformation of on prime applications to transfer to the AWS cloud for Application with include data base migration with AWS cloud automation deployment using DevOps tools. The modernization strategy will include numerous stages. The stages are Analysis & Planning, Data Migration, Extraction &Transform, Quality Engineering and Go-Live/Deployment.

2. G. Ongo and G. P. Kusuma, "Hybrid Database System of MySQL and MongoDB in Web Application Development," 2018 International Conference on Information Management and Technology (ICIMTech), Jakarta, Indonesia, 2018, pp. 256-260, doi: 10.1109/ICIMTech.2018.8528120. keywords: {Time factors;Data models;Object oriented modeling;Database systems;Relational databases;Companies;MySQL;MongoDB;hybrid database;web application;database performance},

In today's business landscape, the exponential growth of data has begun to strain the performance of relational database management systems (RDBMS). Among these systems, MySQL stands as a widely adopted choice, offering reliability and familiarity to users. However, as data volumes escalate, the performance of MySQL can suffer.

To address this challenge, we delve into a comparative analysis focusing solely on MySQL, a mainstay RDBMS solution. We aim to gauge its performance under the weight of increasing data loads, particularly when deployed in a web application context. Our investigation eschews the hybrid approach, concentrating solely on MySQL's capacity to handle high volumes of data.

Our study centers on the development of a simple yet representative web application, simulating a social media platform. This application includes essential features such as user profiles and chat functionalities. By subjecting MySQL to real-world usage scenarios with randomly generated user profiles and chat interactions, we aim to assess its ability to cope with escalating data demands.

Preliminary experiments will be conducted to evaluate MySQL's performance across various metrics, including read and write speeds. These metrics will be scrutinized under increasing data loads to ascertain MySQL's scalability and resilience. Furthermore, we will explore strategies to optimize MySQL's performance, such as indexing and query optimization, to maximize its efficiency in handling large datasets. Through this focused exploration, we seek to provide insights into the real-world performance of MySQL as a standalone database solution for web applications. By elucidating its strengths and limitations in managing high data volumes, we aim to empower businesses with the knowledge needed to make informed decisions regarding their database infrastructure.

3. Dylan Shields, AWS Security , Manning, 2022. keywords: {aws;security;amazon;web;services;iam;indentity;management;access;basics},

Running your systems in the cloud doesn’t automatically make them secure. Learn the tools and new management approaches you need to create secure apps and infrastructure on AWS.

In AWS Security you’ll learn how to:

Securely grant access to AWS resources to coworkers and customers

Develop policies for ensuring proper access controls

Lock-down network controls using VPCs

Record audit logs and use them to identify attacks

Track and assess the security of an AWS account

Counter common attacks and vulnerabilities

Written by security engineer Dylan Shields, AWS Security provides comprehensive coverage on the key tools and concepts you can use to defend AWS-based systems. You'll learn how to honestly assess your existing security protocols, protect against the most common attacks on cloud applications, and apply best practices to configuring identity and access management and virtual private clouds.

NEED FOR PROJECT

In the context of our College Management System project, the need for an efficient and robust database system is paramount. As educational institutions navigate the complexities of modern academia, they generate and manage vast amounts of data encompassing student records, academic programs, faculty information, financial transactions, and more. To effectively manage this wealth of information and support the diverse needs of stakeholders, including students, faculty, administrators, and parents, a reliable database system is essential.

Here's why a solid database system, like MySQL, is crucial for our College Management System project:

1. **Data Management**: A college generates and accumulates a myriad of data points, ranging from student enrollment details to course schedules and academic performance metrics. MySQL provides a structured and organized framework for storing, retrieving, and managing this data efficiently, ensuring its integrity and reliability.

2. **Scalability**: Educational institutions often experience fluctuations in student enrollment, course offerings, and administrative requirements. MySQL's scalability allows our College Management System to adapt seamlessly to changing demands, accommodating growth without compromising performance or data integrity.

3. **Performance**: With a large number of users accessing the system simultaneously, performance is critical for ensuring a smooth and responsive user experience. MySQL's robust architecture and optimization features enable our College Management System to handle high volumes of transactions efficiently, minimizing latency and ensuring timely access to information.

4. **Security**: Protecting sensitive student and institutional data is paramount in today's digital age. MySQL offers robust security features, including access controls, encryption, and authentication mechanisms, to safeguard data against unauthorized access, ensuring compliance with data protection regulations such as GDPR and FERPA.

5. **Integration**: Our College Management System may need to integrate with other systems and tools used by educational institutions, such as learning management systems, student information systems, and financial management software. MySQL's compatibility and support for standard protocols facilitate seamless integration, enabling data sharing and interoperability across different platforms.

6. **Reliability**: Educational institutions rely heavily on the availability and reliability of their database systems to support critical operations, such as student registration, grading, and financial management. MySQL's reputation for stability and uptime ensures uninterrupted access to essential services, minimizing disruptions and downtime.

FEATURES

Certainly! Here are some additional features and aspects related to the College Management System project, including results and exams:

Features of the Project:

1. **Student Management:** The system allows administrators to manage student information, including enrollment, admission, academic records, and personal details. It facilitates the creation and maintenance of comprehensive student profiles, ensuring easy access to relevant information for faculty, administrators, and students themselves.

2. **Faculty Management: Faculty members can utilize the system to manage course materials, assignments, grades, and communication with students. They can upload course materials, grade assignments and exams, and communicate announcements and feedback to students through the platform.

3. **Course Management:** The system supports the creation and management of courses, including course scheduling, curriculum planning, and allocation of resources such as classrooms and teaching materials. Administrators can define course offerings, prerequisites, and enrollment capacities, while faculty members can access course materials and track student progress.

4. **Attendance Tracking:** The system facilitates the tracking of student attendance in classes, lectures, and other academic activities. Faculty members can record attendance, monitor student participation, and generate reports to identify patterns and trends in attendance behavior.

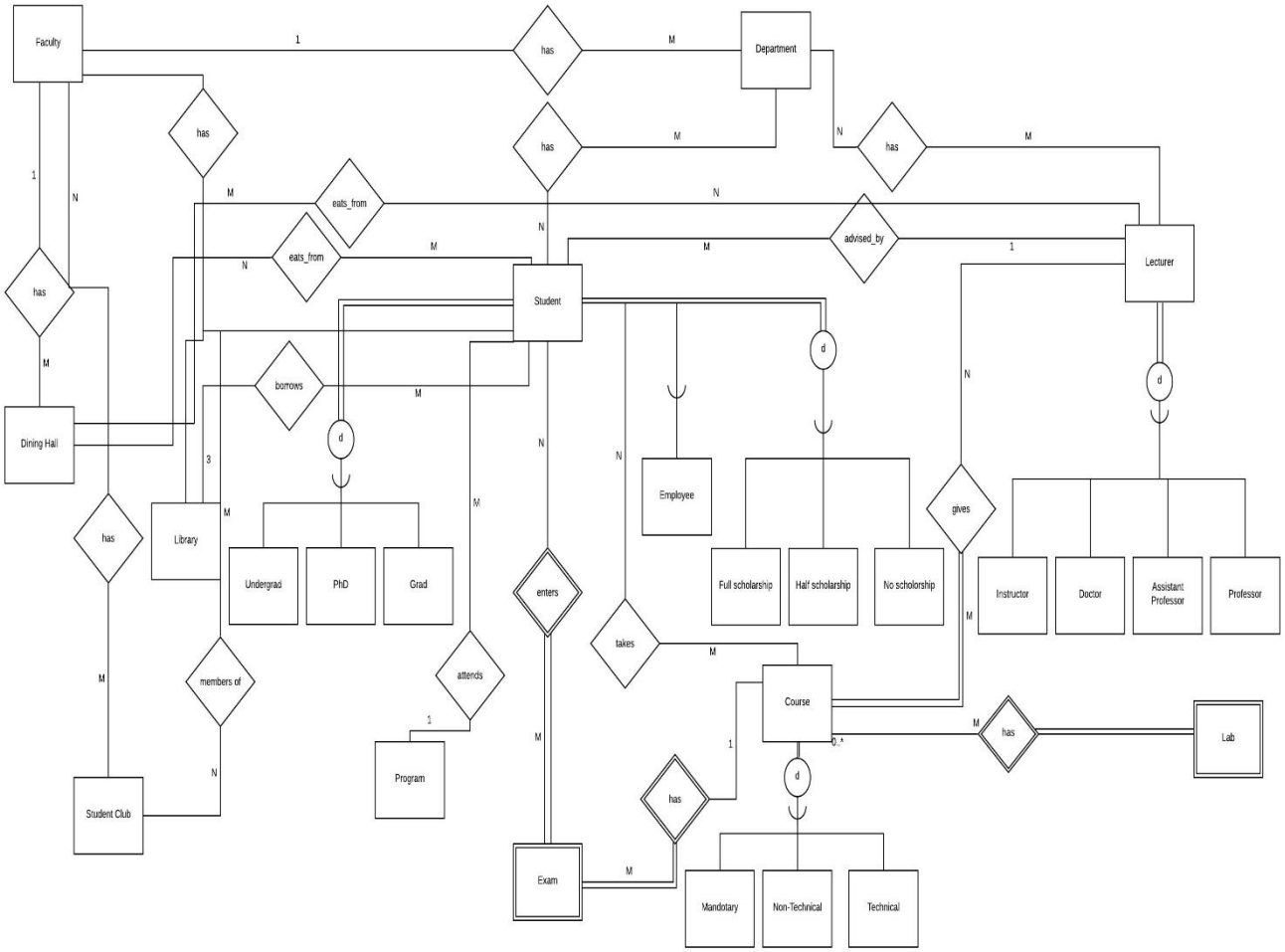
5. **Results Management:** The system manages the recording and processing of

student assessment results, including exams, quizzes, assignments, and projects. It allows faculty members to input grades, calculate final grades, and generate transcripts and academic reports for students.

6. **Examination Management:** The system supports the creation, scheduling, and administration of exams, including midterms, finals, and other assessments. Administrators can define exam schedules, assign exam locations, and manage exam logistics, while faculty members can create exam papers, grade responses, and analyze results.

8. **Financial Management:** The system includes features for managing financial transactions, such as tuition payments, fee collections, and budget allocations. It allows administrators to track financial transactions, generate invoices, and manage accounts receivable and payable, ensuring accurate and transparent financial records.

ER DIAGRAM


Faculty
has
Department
has
has
has
eats_from
eats_from
advised_by
Lecturer
Student
has
Dining Hall
borrows
d
Employee
Full scholarship
Half scholarship
No scholorship
gives
Instructor
Doctor
Assistant Professor
Professor
has
Library
Undergrad
PhD
Grad
enters
takes
Course
has
Lab
members of
attends
Program
Student Club
has
Exam
Mandotary
Non-Technical
Technical

EXPLANATION

Sure, let's go through each entity in the database schema and explain their purpose:

1. **Book:** This entity represents books available in the library.
   - `library_name`: Name of the library where the book is available.
   - `book_name`: Title of the book.
   - `author`: Author(s) of the book.
   - `book_status`: Indicates whether the book is available or unavailable.

2**. Course:** This entity contains information about academic courses offered by the institution.
   - `course_name`: Name of the course.
   - `AKTS`: AKTS (European Credit Transfer and Accumulation System) credits for the course.
   - `class_no`: Class number or code for the course.
   - `course_type`: Type of course (e.g., technical, non-technical).
   - `lecturer`: Name of the lecturer teaching the course.
   - `prerequisite`: Prerequisite course(s) required for enrollment.
   - `semester`: Semester in which the course is offered.

3. **Dining:** Represents daily dining menus available in the dining hall or cafeteria.
   - `date`: Date of the dining menu.
   - `dining_name`: Name or location of the dining facility.
   - `soup`: Option for soup available in the menu.
   - `main_dish`: Main dish available in the menu.
   - `side_dish`: Side dish available in the menu.
   - `dessert`: Dessert option available in the menu.

4**. Exam**:** This entity stores details about exams associated with specific courses.
   - `course_name`: Name of the course for which the exam is conducted.
   - `exam_name`: Name or type of the exam (e.g., midterm, final).

5**. Lecturer**:** Contains information about lecturers or faculty members.
   - `lecturer_name`: Name of the lecturer.
   - `website`: Website or profile link of the lecturer.
   - `birthdate`: Birthdate of the lecturer.
   - `email`: Email address of the lecturer.
   - `degree`: Academic degree or title of the lecturer.
   - `role`: Role or position of the lecturer.
   - `course_name`: Name of the course(s) taught by the lecturer.

- `advised_st_id`: Student ID of the advised student (if applicable).
- `department`: Department to which the lecturer belongs.

6. **Program****: Stores information about academic programs or exchange programs.
- `program_name`: Name of the program.
- `est_semester_duration`: Estimated duration of the program in semesters.
- `student_id`: Student ID associated with the program.
- `other_uni_name`: Name of another university (if applicable, e.g., for exchange programs).

These entities capture essential aspects of the college management system, including library resources, academic courses, dining services, exams, faculty members, and academic programs. They facilitate data organization, retrieval, and management for efficient operation and decision-making within the educational institution.

CARDINALITY –

To explain cardinalities for the entities in the provided database schema, let's examine each table and the relationships between them:

1. Book:

- `library_name` to `book_name`: One library can have many books, so it's a one-to-many relationship.

- `book_name` to `author`: One book can have only one author, so it's a one-to-one relationship.

- `book_name` to `book_status`: One book can have only one status, so it's a one-to-one relationship.

2. Course:

- `course_name` to `AKTS`: One course can have only one AKTS value, so it's a one-to-one relationship.

- `course_name` to `class_no`: One course can have only one class number, so it's a one-to-one relationship.

- `course_name` to `course_type`: One course can have only one course type, so it's a one-to-one relationship.

- `course_name` to `lecturer`: One course can have only one lecturer, so it's a one-to-one relationship.

- `course_name` to `prerequisite`: One course can have only one prerequisite, so it's a one-to-one relationship.

- `course_name` to `semester`: One course can have only one semester, so it's a one-to-one relationship.

3. Dining:

- `date` to `dining_name`: One dining date can have many dining names, so it's a one-to-many relationship.

- `dining_name` to `soup`, `main_dish`, `side_dish`, `dessert`: Each dining name has only one soup, main dish, side dish, and dessert, so they're all one-to-one relationships.

4. Exam:

- `course_name` to `exam_name`: One course can have many exams, so it's a one-to-many relationship.

5. Lecturer:

- `lecturer_name` to `email`: One lecturer can have only one email, so it's a one-to-one relationship.

- `course_name` to `lecturer`: One course can have only one lecturer, so it's a one-to-one relationship.

6. Program:

- `program_name` to `est_semester_duration`: One program can have only one estimated semester duration, so it's a one-to-one relationship.

- `program_name` to `student_id`: One program can have many students, so it's a one-to-many relationship.

7. Student:

- `student_id` to `name`, `birthdate`, `course_code`, `advisor_name`, `degree`, `scholarship`, `grade`, `emp_type`, `department`, `email`: Each student ID has only one set of these attributes, so they're all one-to-one relationships.

8. Student_book:

- `student_id` to `book_name`: One student can borrow many books, and one book can be borrowed by many students, so it's a many-to-many relationship.

9. Student_club:

- `club_name` to `leader_id`: One club can have only one leader, so it's a one-to-one relationship.

10. Student_club_participants:

- `leader_id` to `participant_id`: One leader can have many participants, and one participant can participate in many clubs, so it's a many-to-many relationship.

11. Student_current_course:

- `course_name` to `student_id`: One course can have many students, and one student can take many courses, so it's a many-to-many relationship.

12. Student_exam:

- `exam_name` to `student_id`: One exam can have many students, and one student can take many exams, so it's a many-to-many relationship.

13. Student_past_courses:

- `student_id` to `course_name`: One student can have many past courses, and one past course can belong to many students, so it's a many-to-many relationship.

14. Visitor:

- `email` to `name`, `password`, `phone`, `type`: Each email has only one set of these attributes, so they're all one-to-one relationships.

These cardinalities describe the relationships between the entities in the database schema.

RELATIONAL TABLE

TABLE BOOK AND COURSE

TABLE – 'book'

Library_name	Primary Key, Varchar(255)
Book_name	Primary key, Varchar(200)
Author	Primary key, Varchar(150)
Book_status	Varchar(255)

TABLE – 'course'

Course_name	Primary key, Varchar(255)
AKTS	Int(255)
Class_no	Varchar(255)
Course_type	Varchar(255)
lecturer	varchar(255)
prerequisite	varchar(255)
semester	varchar(255)

TABLE - lecturer

lecturer_name	varchar(255)
Website	varchar(255)
Birthdate	date
email	Primary Key, varchar(255)
degree	varchar(255)
role	varchar(255)
course_name	varchar(255)
advised_st_id	Int(255)
department	varchar(255)

TABLE - program

program_name	Primary Key, varchar(255)
est_semester_duration	Int(255)
student_id	Int(255)
other_uni_name	varchar(255)

TABLE - dining

date	Primary key, varchar(255)
Dining_name	Primary key, varchar(255)
soup	varchar(255)
Main_dish	varchar(255)
Side_dish	varchar(255)
dessert	varchar(255)

TABLE - exam

Course_name	Primary key, varchar(255)
Exam_name	Primary Key, varchar(255)

TABLE – student_exaqm

exam_name	Primary Key, varchar(255)
student_id	Primary Key, int(255)
grade	int(255)

TABLE – student_past_courses

student_id	Primary Key, int(255)
course_name	Primary Key, varchar(255)
final_grade	int(255)

TABLE - visitor

email	Primary Key, varchar(255)
Name	varchar(255)
Password	varchar(255)
Phone	varchar(255)
type	varchar(255)

TABLE - student

student_id	Primary Key, int(255)
name	varchar(255)
birthdate	Date
course_code	varchar(255)
advisor_name	varchar(255)
degree	varchar(255)
scholarship	varchar(255)
grade	Int(255)
emp_type	varchar(255)
Department	varchar(255)
email	varchar(255)

TABLE – student_book

student_id	Primary Key, Int(255)
book_name	Primary Key, varchar(255)

TABLE – student_club

club_name	Primary Key, varchar(255)
leader_id	int(255)

TABLE – student_club_participants

leader_id	Primary key, int(11)
participant_id	Primary Key, int(11)

TABLE – student_current_course

course_name	Primary Key, varchar(255)
student_id	int(255)

EXPLANATION

Book table:

Data Types: library_name varchar (255), book_name varchar(255), author varchar(255), book_status varchar(255)

Primary Key: (library_name, book_name, author)

Foreign Keys: None

course table:

Data Types: course_name varchar (255), AKTS int(255), class_no varchar(255), course_type varchar(255), lecturer varchar(255), prerequisite varchar(255), semester varchar(255)

Primary Key: (course_name)

Foreign Keys: lecturer.course_name references course.course_name

dining table:

Data Types: date date , dining_name varchar(255), soup varchar(255), main_dish varchar(255), side_dish varchar(255), dessert varchar(255)

Primary Key: (date, dining_name)

Foreign Keys: None

exam table:

Data Types: course_name varchar (255), exam_name varchar(255)

Primary Key: (course_name, exam_name)

Foreign Keys: None

lecturer table:

Data Types: lecturer_name varchar (255), website varchar(255), birthdate date, email varchar(255), degree varchar(255), role varchar(255), course_name varchar(255), advised_st_id int(255), department varchar(255)

Primary Key: (email)

Foreign Keys: None

program table:

Data Types: program_name varchar(255), est_semester_duration int(255), student_id int(255), other_uni_name varchar(255)

Primary Key: (program_name)

Foreign Keys: None

student table:

Data Types: student_id int(255), name varchar(255), birthdate date, course_code varchar(255), advisor_name varchar(255), degree varchar(255), scholarship varchar(255), grade int(255), emp_type varchar(255), department varchar(255), email varchar(255)

Primary Key: (student_id)

Foreign Keys: lecturer.advisor_name references student.name, student_book.student_id references student.student_id, student_club_participants.participant_id references student.student_id, student_current_course.student_id references student.student_id, student_exam.student_id references student.student_id, student_past_courses.student_id references student.student_id

student_book table:

Data Types: student_id int(255), book_name varchar(255)

Primary Key: (student_id, book_name)

Foreign Keys: book.book_name references student_book.book_name

student_club table:

Data Types: club_name varchar(255), leader_id int(255)

Primary Key: (club_name)

Foreign Keys: None

student_club_participants table:

Data Types: leader_id int(11), participant_id int(11)

Primary Key: (leader_id, participant_id)

Foreign Keys: student.student_id references student_club_participants.participant_id

student_current_course table:

Data Types: course_name varchar(255), student_id int(255)

Primary Key: (course_name)

Foreign Keys: course.course_name references student_current_course.course_name, student.student_id references student_current_course.student_id

student_exam table:

Data Types: exam_name varchar(255), student_id int(255), grade int(255)

Primary Key: (exam_name, student_id)

Foreign Keys: exam.exam_name references student_exam.exam_name

student_past_courses table:

Data Types: student_id int(255), course_name varchar(255), final_grade int(255)

Primary Key: (student_id, course_name)

Foreign Keys: course.course_name references student_past_courses.course_name

visitor table:

Data Types: email varchar(255), name varchar(255), password varchar(255), phone varchar(255), type varchar(255)

Primary Key: (email)

Foreign Keys: None

SQL QUERRIES

1. User Query: "List all courses available in the Spring semester."

```
SELECT *
FROM course
WHERE semester = 'Spring';
```

2.User Query: "Find all courses taught by Reda Alhajj."

```
SELECT *
FROM course
WHERE lecturer = 'Reda Alhajj';
```

3.User Query: "Retrieve the dining menu for June 20, 2020, at the North Campus Dining Hall

```
SELECT *
FROM dining
WHERE date = '2020-06-20' AND dining_name = 'North Campus Dining Hall';
```

4.User Query: "List all students who are club leaders."

```
SELECT *
FROM student
WHERE student_id IN (
    SELECT leader_id
    FROM student_club
```

5.User Query: "Find the average grade of students in the Algorithm course."

```
SELECT AVG(grade) AS average_grade
FROM student_exam
WHERE exam_name IN (
    SELECT exam_name
    FROM exam
    WHERE course_name = 'Algorithm'
);
```

6.User Query: "Retrieve all students who have borrowed books but haven't returned them."

```
SELECT *
FROM student
WHERE student_id IN (
    SELECT student_id
    FROM student_book
```

7.User Query: "List all lecturers along with their email addresses

```
SELECT lecturer_name, email
FROM lecturer;
```

8.User Query: "Find all students who have a scholarship and are in the Computer Science & Engineering department."

```
SELECT *
FROM student
WHERE scholarship IS NOT NULL AND department = 'Computer Science & Engineering';
```

9.User Query: "Retrieve the names of students who have borrowed a specific book."

```
SELECT s.name AS student_name
FROM student s
INNER JOIN student_book sb ON s.student_id = sb.student_id
WHERE sb.book_name = 'Harry Potter';
```

10.User Query: "Find all lecturers who are also advisors for students."

```
SELECT *
FROM lecturer
WHERE lecturer_name IN (
    SELECT DISTINCT advisor_name
    FROM student
```

ARCHITECTURE DESIGN

Entity-Relationship Model:

The design follows an entity-relationship model where tables represent entities, and relationships are established through keys.

Primary keys uniquely identify records in each table, while foreign keys establish relationships between tables.

Tables and Entities:

book Table:

Represents information about books in a library, including title, author, and availability status.

course Table:

Contains details about academic courses such as name, AKTS (European Credit Transfer System), lecturer, etc.

dining Table:

Stores information about daily dining menus, including date, dining hall, and various dishes.

exam Table:

Represents exams associated with specific courses.

lecturer Table:

Contains information about lecturers, including their name, email, and associated courses.

program Table:

Represents academic programs with details about the program name, estimated semester duration, and participating students.

student Table:

Contains information about students, including their ID, name, birthdate, degree, scholarship, and more.

student_book Table:

Represents a many-to-many relationship between students and books, indicating which books are associated with each student.

student_club Table:

Stores information about student clubs and their leaders.

student_club_participants Table:

Establishes a many-to-many relationship between students and clubs, indicating which students participate in each club.

student_current_course Table:

Indicates the courses that students are currently enrolled in.

student_exam Table:

Represents the exams taken by students, including their grades.

student_past_courses Table:

Stores information about courses that students have completed in the past, along with their final grades.

visitor Table:

Represents visitors, including details like email, name, password, phone, and type.

Key Design Features:

Primary Keys:

Primary keys uniquely identify records in each table.

Composite primary keys are used in some tables, such as (library_name, book_name, author).

Foreign Keys:

Foreign keys establish relationships between tables. For example, the course table's lecturer column references the email column in the lecturer table.

Normalization:

The design appears to follow normalization principles to reduce redundancy and improve data integrity.

Data Types:

Appropriate data types are assigned to each column, ensuring data accuracy and efficient storage.

Overall Architecture:

The architecture is relational, emphasizing the relationships between different entities in a structured manner.

Relationships are established through foreign keys, supporting referential integrity.

Usage Scenarios:

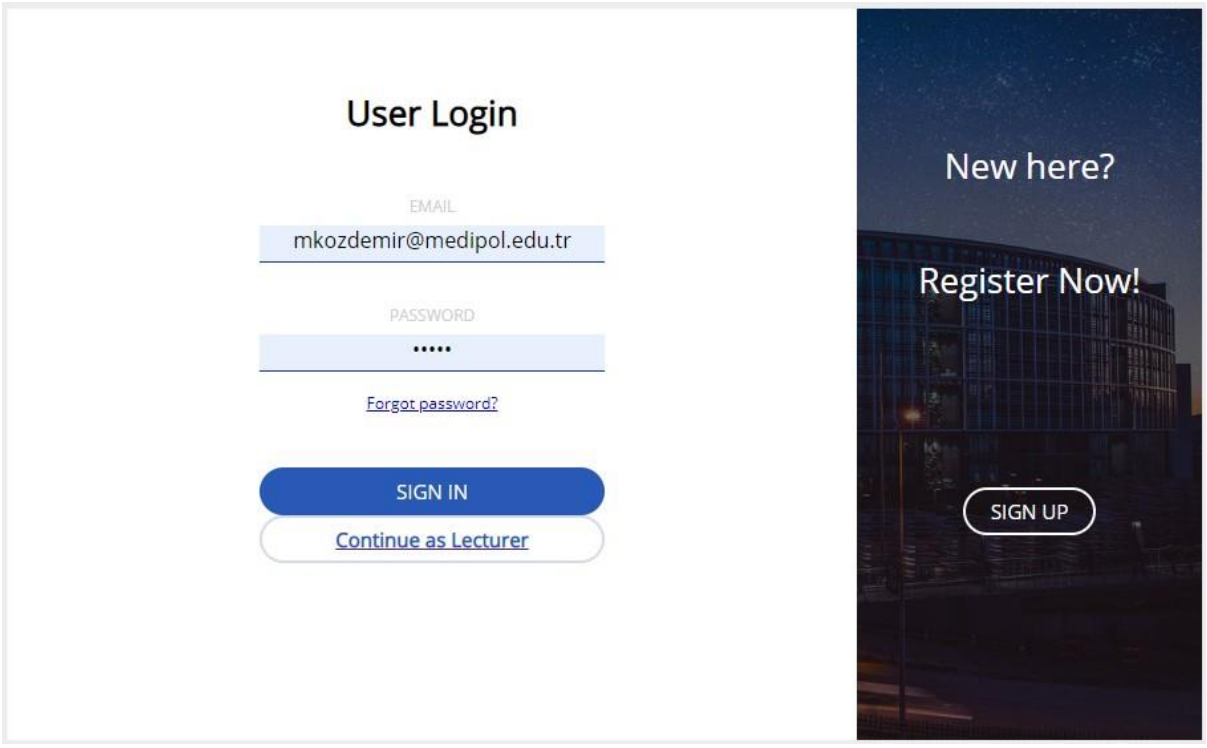
The schema is designed to support a variety of scenarios, including library management, academic course tracking, dining services, student club participation, and more.

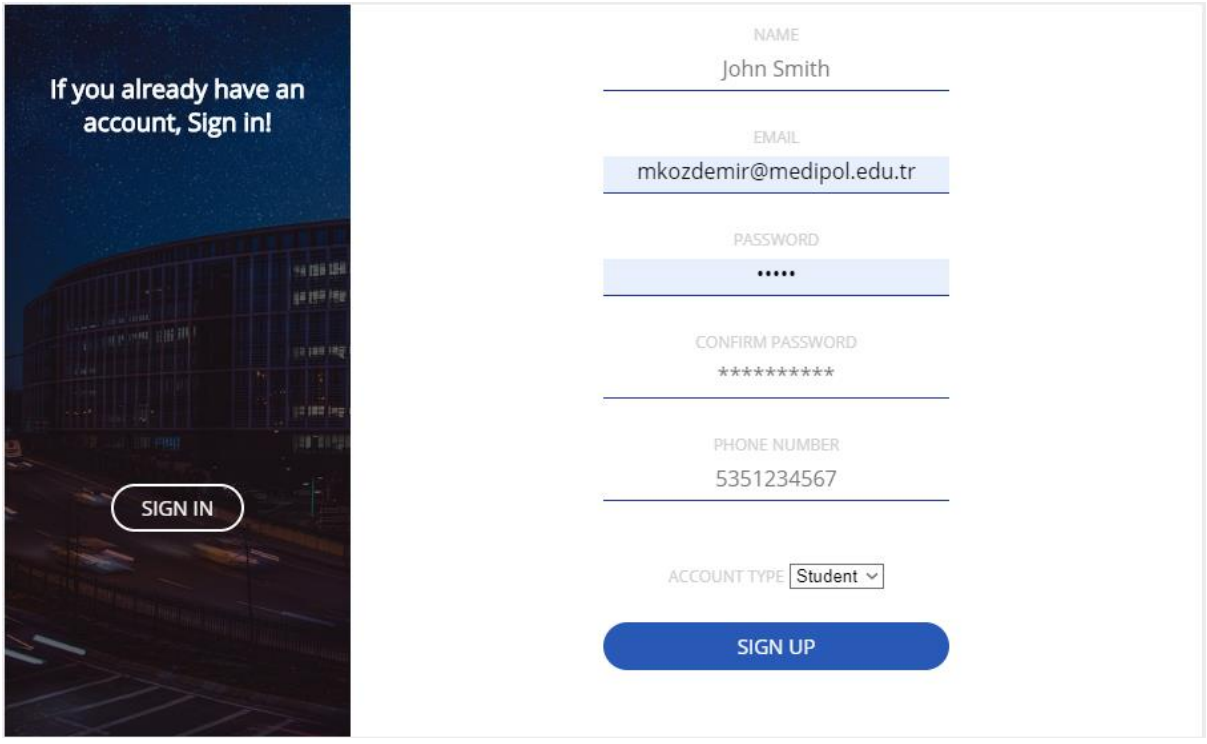
Considerations:

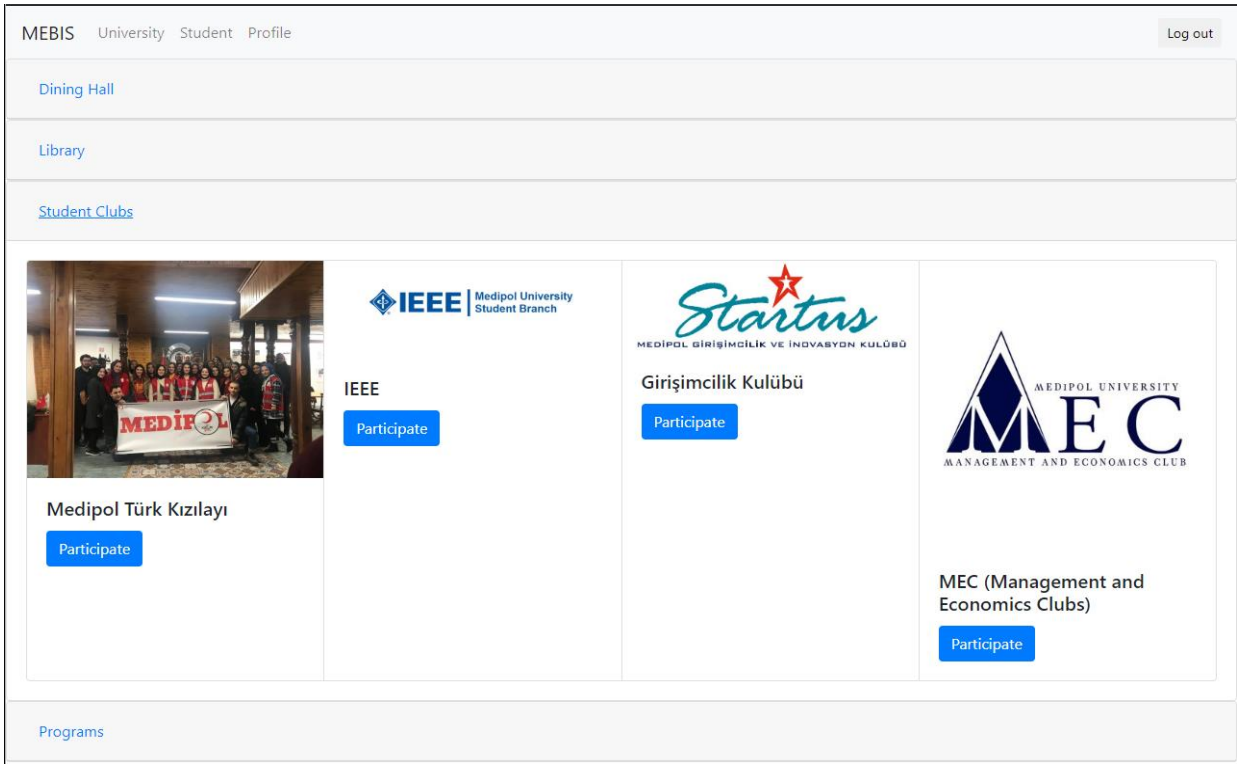
The design could be further optimized based on specific use cases and performance requirements.

Indexing strategies should be considered for efficient query performance.

WEBPAGE


User Login
EMAIL
mkozdemir@medipol.edu.tr
PASSWORD
•••••
Forgot password?
SIGN IN
Continue as Lecturer
New here?
Register Now!
SIGN UP

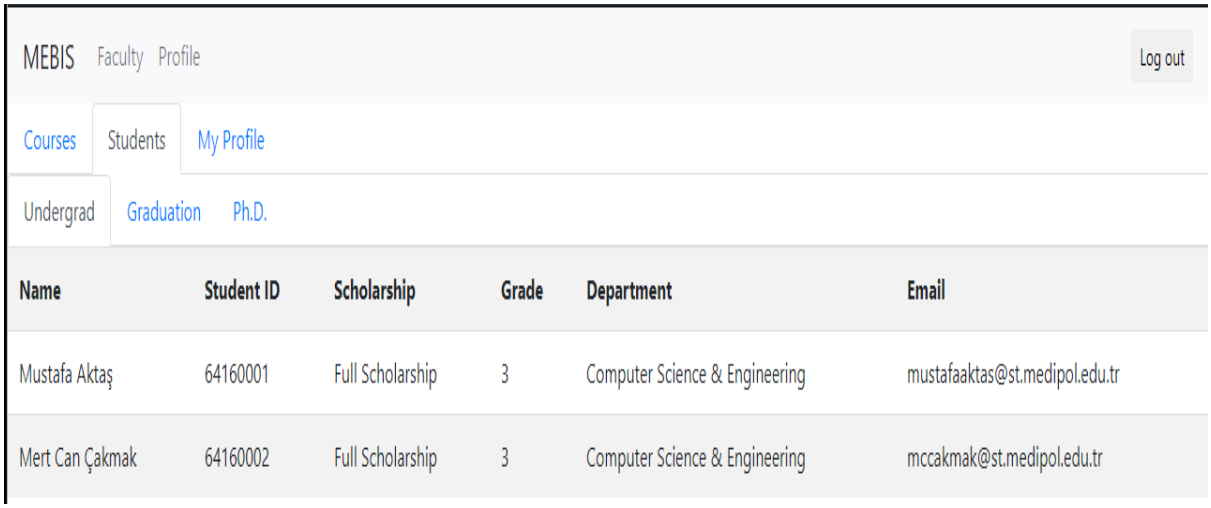

If you already have an account, Sign in!
SIGN IN
NAME
John Smith
EMAIL
mkozdemir@medipol.edu.tr
PASSWORD
•••••
CONFIRM PASSWORD
****
PHONE NUMBER
5351234567
ACCOUNT TYPE
Student
SIGN UP



MEBIS University Student Profile Log out

Past Courses | Courses | Exam Results | Course Selection

Course Name	AKTS	Lecturer	Course Type	Status
Calculus 1	8	Mehmed Rafet ÖZDEMİR	Technical	-
Calculus 2	8	Mehmed Rafet ÖZDEMİR	Technical	Calculus 1
Database	8	Reda Alhajj	Technical	-
Machine Learning	6	Bahadır Kürşat Güntürk	Non-Technical	-
Microprocessor	6	Mehmet Kocatürk	Technical	-
Probability	8	Mehmet Kemal Özdemir	Technical	Calculus 2
Web Design	8	Muhsin Zahid Uğur	Non technical	-



CONCLUSION:

In conclusion, the College Management System (CMS) offers a transformative solution for academic institutions seeking to modernize and streamline their operations. By leveraging cutting-edge technology and innovative design principles, the CMS enables colleges and universities to enhance their administrative efficiency, improve communication, and create personalized learning experiences for students.

The emphasis on accessibility and inclusivity ensures that the CMS is a tool for all, providing multi-modal content formats and integrating assistive technologies to meet the needs of diverse learners. Its robust architecture and scalability make it suitable for institutions of all sizes, allowing for flexibility and growth as the educational landscape evolves.

Through adaptive learning algorithms, personalized recommendations, and comprehensive tracking of student progress, the CMS fosters an environment that nurtures student success. The advanced analytics and resource allocation features equip educational institutions with the tools needed to make informed decisions and drive efficiency.

Ultimately, our CMS is a pivotal solution that redefines the operational paradigms of modern education, setting a new standard for efficiency, collaboration, and excellence. By focusing on both current and future needs, it empowers academic institutions to embrace change and achieve unprecedented success.

GITHUB LINK:

https://github.com/NoorieSideEye/DBMS-PROJECT